

# CES-35 – Redes de Computadores e Internet

## Laboratório 2: Construindo um Simples Servidor de Arquivo

**Nomes:** Daniel Araujo Cavassani (server) | Caue Marçal Guimarães (client) | (COMP 25)

**Data:** 09/09/2024

### 1. A Solução

Neste laboratório, desenvolvemos um sistema cliente-servidor de arquivos usando soquetes em linguagem C, sem o uso de protocolos de aplicação externos, como HTTP. O protocolo implementado permite que o cliente faça requisições de arquivo e verifique o último acesso ao servidor.

### Protocolo Implementado

O protocolo projetado segue as seguintes características:

- O cliente envia um comando **MyGet <nome\_do\_arquivo>** ao servidor, solicitando um arquivo específico. O servidor responde enviando o conteúdo do arquivo solicitado, ou uma mensagem de erro caso o arquivo não seja encontrado.
- O cliente também pode enviar o comando **MyLastAccess**, solicitando ao servidor o último instante de acesso realizado por este cliente. Caso seja a primeira requisição, o servidor retorna "Last Access=NULL".
- Implementamos um mecanismo de estado parcial no servidor, onde ele mantém o histórico de acessos dos clientes para fornecer a informação do último acesso.

A comunicação é feita sobre TCP, com a criação de soquetes para troca de mensagens.

# Estrutura do Código

## Servidor (server.cpp)

O código do servidor foi implementado utilizando threads para lidar com múltiplos clientes simultaneamente. Para garantir que cada cliente seja atendido sem bloqueios, cada nova conexão inicia uma nova thread para processar as requisições.

Cada cliente recebe uma cor diferente no terminal para facilitar o monitoramento das conexões, e os comandos e respostas são exibidos com o IP e a porta de origem do cliente. O histórico de acessos é mantido em um array de clientes, e esse histórico é atualizado a cada requisição.

Pontos de destaque:

- Cada cliente é identificado por uma combinação de IP e porta. O servidor mantém o histórico de clientes e suas respectivas cores para facilitar o monitoramento.
- O servidor mantém a conexão aberta até que o cliente envie o comando `exit()`, permitindo múltiplas requisições sem a necessidade de reconexão.
- A funcionalidade de `print` periódico exibe o histórico de clientes conectados e seus respectivos últimos acessos a cada 10 segundos.

## Client (client.cpp)

O cliente é responsável por enviar comandos para o servidor e receber os arquivos ou dados de acesso. O cliente também permite a especificação de uma porta de origem, mas usa uma porta efêmera por padrão.

Comandos Implementados:

- **MyGet <arquivo>**: Solicita o conteúdo de um arquivo ao servidor.
- **MyLastAccess**: Solicita a informação do último acesso ao servidor.
- **exit()**: Fecha a conexão com o servidor.

## Link para o Código e Reprodutibilidade

O código completo do cliente e servidor pode ser acessado no [repositório](#). Para reproduzir o experimento:

1. Compile o servidor com: **gcc -o server server.cpp -lpthread**
2. Compile o cliente com: **gcc -o client client.cpp**
3. Execute o servidor: **./server**
4. Execute o cliente: **./client <endereço\_do\_servidor> [porta\_opcional]**

## 2. Testes

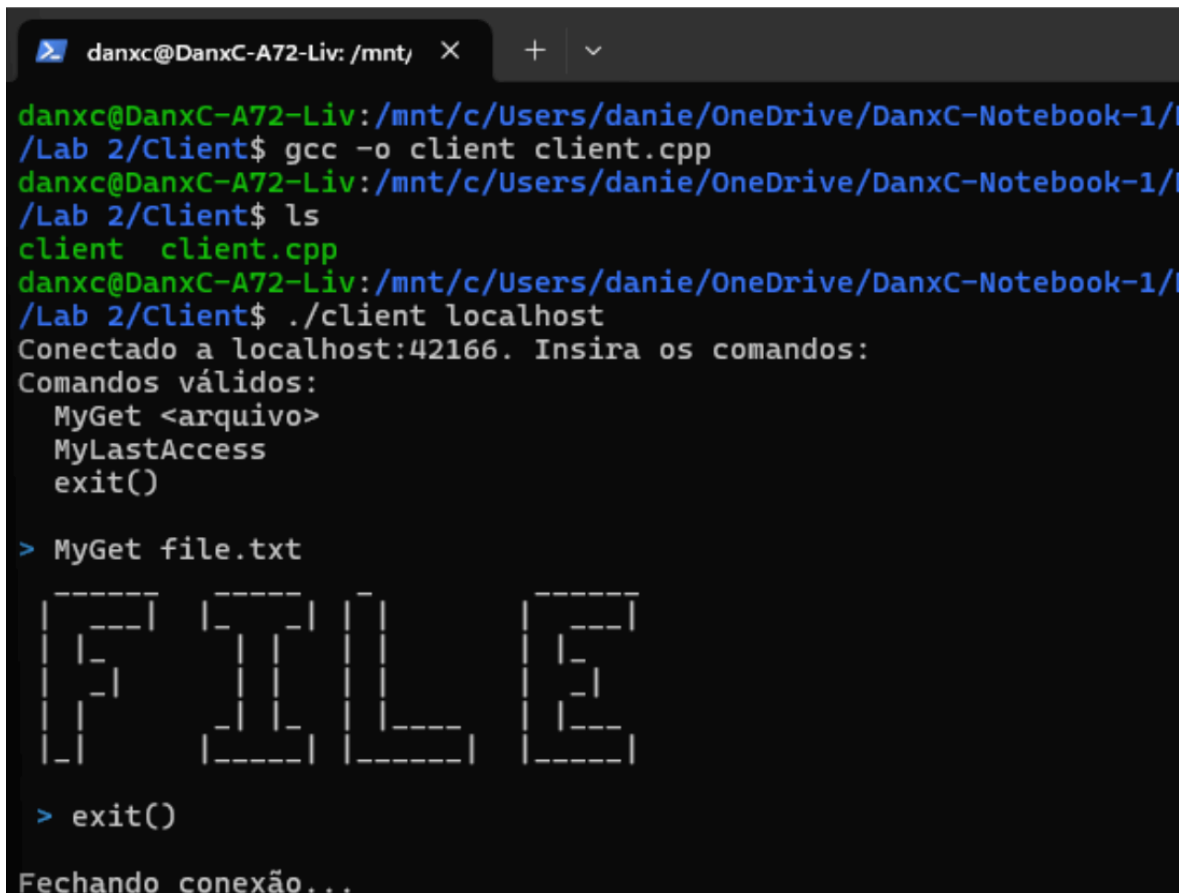
### Testes Realizados

Para garantir que o sistema está funcionando corretamente, realizamos diversos testes com múltiplos clientes conectados ao mesmo tempo. Abaixo, descrevemos os cenários testados:

#### Cenário 1: Requisição de arquivo válida

- O cliente enviou o comando MyGet file.txt.
- O servidor encontrou o arquivo e enviou o conteúdo com sucesso.
- O cliente exibiu corretamente o conteúdo recebido.

client



```
danxc@DanxC-A72-Liv: /mnt/ X + v
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Lab 2/Client$ gcc -o client client.cpp
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Lab 2/Client$ ls
client client.cpp
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Lab 2/Client$ ./client localhost
Conectado a localhost:42166. Insira os comandos:
Comandos válidos:
  MyGet <arquivo>
  MyLastAccess
  exit()

> MyGet file.txt

FILE

> exit()
Fechando conexão...
```

server

```
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Documentos/Academi
/Lab 2/Server$ gcc -o server server.cpp -lpthread
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Documentos/Academi
/Lab 2/Server$ ls
file.txt  server  server.cpp
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/Documentos/Academi
/Lab 2/Server$ ./server
Server is listening on port 8080

----- Client Histories -----
-----

(127.0.0.1:42166) Client connected.

----- Client Histories -----
-----

----- Client Histories -----
-----

(127.0.0.1:42166) Received command: MyGet file.txt
(127.0.0.1:42166) Requested file: file.txt

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
-----

(127.0.0.1:42166) Connection closed.

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
```

## Cenário 2: Requisição de arquivo inexistente

- O cliente enviou o comando MyGet arquivo\_invalido.txt.
- O servidor não encontrou o arquivo e enviou uma mensagem de erro: Error: File not found.
- O cliente exibiu a mensagem de erro.

client

```
danxc@DanxC-A72-Liv: /mnt/ X + v
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/
/Lab 2/Client$ ./client localhost
Conectado a localhost:43454. Insira os comandos:
Comandos válidos:
  MyGet <arquivo>
  MyLastAccess
  exit()

> MyGet arquivo_invalido.txt
Error: File not found
> |
```

server

```
(127.0.0.1:43454) Client connected.

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
-----

(127.0.0.1:43454) Received command: MyGet arquivo_invalido.txt
(127.0.0.1:43454) Requested file: arquivo_invalido.txt

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
-----
```

### Cenário 3: Verificação de último acesso

- O cliente enviou o comando MyLastAccess logo após conectar.
- O servidor respondeu Last Access=NULL, pois era a primeira requisição do cliente.
- O cliente enviou novamente o comando após uma nova requisição, e o servidor respondeu com o horário correto do último acesso.

client

```
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC  
/Lab 2/Client$ ./client localhost  
Conectado a localhost:58034. Insira os comandos:  
Comandos válidos:  
  MyGet <arquivo>  
  MyLastAccess  
  exit()  
  
> MyLastAccess  
Last Access=NULL  
> MyLastAccess  
Last Access=2024-09-09 22:33:18  
>
```

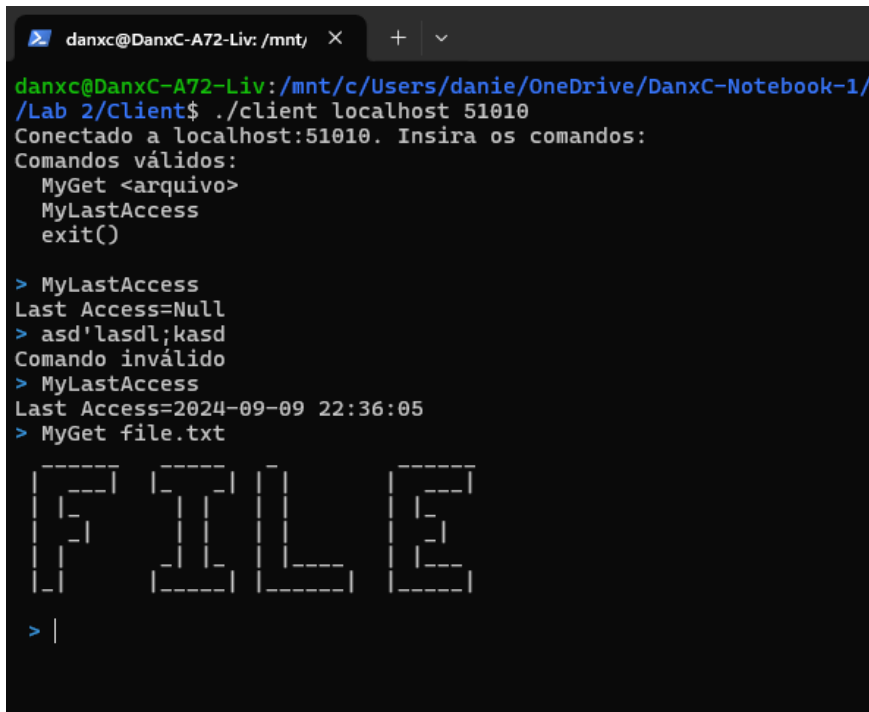
server

```
(127.0.0.1:58034) Client connected.  
  
----- Client Histories -----  
Client 127.0.0.1:42166, Last Access: 1725931472  
Client 127.0.0.1:43454, Last Access: 1725931667  
Client 127.0.0.1:33996, Last Access: 1725931970  
-----  
  
(127.0.0.1:58034) Received command: MyLastAccess  
  
----- Client Histories -----  
Client 127.0.0.1:42166, Last Access: 1725931472  
Client 127.0.0.1:43454, Last Access: 1725931667  
Client 127.0.0.1:33996, Last Access: 1725931970  
Client 127.0.0.1:58034, Last Access: 1725931998  
-----  
  
(127.0.0.1:58034) Received command: MyLastAccess
```

## Cenário 4: Múltiplos clientes simultâneos

- Conectamos três clientes simultaneamente ao servidor.
- Cada cliente fez requisições diferentes (MyGet, MyLastAccess, etc.).
- O servidor atendeu cada cliente corretamente em threads separadas, sem bloqueios ou interrupções.
- O histórico de clientes foi atualizado e printado corretamente.

client 1

A terminal window titled 'danxc@DanxC-A72-Liv: /mnt/' shows the execution of a client program. The user runs './client localhost 51010', which connects to 'localhost:51010'. The program prompts for valid commands: 'MyGet <arquivo>', 'MyLastAccess', and 'exit()'. The user enters 'MyLastAccess', and the program returns 'Last Access=NULL'. Then, the user enters 'asd\'lasdl;kasd', which is marked as 'Comando inválido'. Next, the user enters 'MyLastAccess' again, and the program returns 'Last Access=2024-09-09 22:36:05'. Finally, the user enters 'MyGet file.txt', and the program outputs 'FILE' in a large, stylized, dashed font. The prompt '> |' is visible at the bottom.

```
danxc@DanxC-A72-Liv: /mnt/ X + v
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1/
/Lab 2/Client$ ./client localhost 51010
Conectado a localhost:51010. Insira os comandos:
Comandos válidos:
  MyGet <arquivo>
  MyLastAccess
  exit()

> MyLastAccess
Last Access=NULL
> asd'lasdl;kasd
Comando inválido
> MyLastAccess
Last Access=2024-09-09 22:36:05
> MyGet file.txt
FILE
> |
```

client 2

```
danxc@DanxC-A72-Liv: /mnt/ X + v
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1
/Lab 2/Client$ ./client localhost 51011
Conectado a localhost:51011. Insira os comandos:
Comandos válidos:
  MyGet <arquivo>
  MyLastAccess
  exit()

> MyGet file.txt

FILE

> MyLastAccess
Last Access=2024-09-09 22:36:10
> MyGet file.txt

FILE

> MyLastAccess
Last Access=2024-09-09 22:36:56
>
```

client 3

```
danxc@DanxC-A72-Liv: /mnt/ X + v
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1
/Lab 2/Client$ 51012
51012: command not found
danxc@DanxC-A72-Liv:/mnt/c/Users/danie/OneDrive/DanxC-Notebook-1
/Lab 2/Client$ ./client localhost 51012
Conectado a localhost:51012. Insira os comandos:
Comandos válidos:
  MyGet <arquivo>
  MyLastAccess
  exit()

> MyLastAccess
Last Access=NULL
> daskasfkofapsokapsok
Comando inválido
> MyLastAccess
Last Access=2024-09-09 22:36:25
> MyLastAccess
Last Access=2024-09-09 22:36:47
>
```



server

```
(127.0.0.1:51010) Client connected.

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932165
Client 127.0.0.1:51011, Last Access: 1725932170
-----

(127.0.0.1:51010) Received command: MyLastAccess
(127.0.0.1:51011) Client connected.
(127.0.0.1:51011) Received command: MyGet file.txt
(127.0.0.1:51011) Requested file: file.txt

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932165
Client 127.0.0.1:51011, Last Access: 1725932170
-----

(127.0.0.1:51012) Client connected.

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932165
Client 127.0.0.1:51011, Last Access: 1725932170
-----

(127.0.0.1:51012) Received command: MyLastAccess

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932165
Client 127.0.0.1:51011, Last Access: 1725932170
Client 127.0.0.1:51012, Last Access: 1725932185
-----
```

```
----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932165
Client 127.0.0.1:51011, Last Access: 1725932170
Client 127.0.0.1:51012, Last Access: 1725932185
-----

(127.0.0.1:51010) Received command: MyLastAccess

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932198
Client 127.0.0.1:51011, Last Access: 1725932170
Client 127.0.0.1:51012, Last Access: 1725932185
-----

(127.0.0.1:51011) Received command: MyLastAccess
(127.0.0.1:51012) Received command: MyLastAccess
(127.0.0.1:51010) Received command: MyGet file.txt
(127.0.0.1:51010) Requested file: file.txt

----- Client Histories -----
Client 127.0.0.1:42166, Last Access: 1725931472
Client 127.0.0.1:43454, Last Access: 1725931667
Client 127.0.0.1:33996, Last Access: 1725931970
Client 127.0.0.1:58034, Last Access: 1725932008
Client 127.0.0.1:51010, Last Access: 1725932212
Client 127.0.0.1:51011, Last Access: 1725932204
Client 127.0.0.1:51012, Last Access: 1725932207
-----

(127.0.0.1:51011) Received command: MyGet file.txt
(127.0.0.1:51011) Requested file: file.txt
(127.0.0.1:51011) Received command: MyLastAccess
(127.0.0.1:51012) Received command: MyLastAccess
```

Server\_client1\_client2\_client3

The image displays four terminal windows from a Metasploit Meterpreter session, illustrating the process of connecting to a remote host and retrieving a file.

- Top-Left Window:** Shows the initial connection to 10.10.10.10. The user enters `file.txt` in the `MyGet` command, and the file content is displayed as `FILE`.
- Top-Right Window:** Shows the user entering `file.txt` in the `MyGet` command. The file content is displayed as `FILE`.
- Bottom-Left Window:** Shows the user entering `file.txt` in the `MyGet` command. The file content is displayed as `FILE`.
- Bottom-Right Window:** Shows the user entering `file.txt` in the `MyGet` command. The file content is displayed as `FILE`.